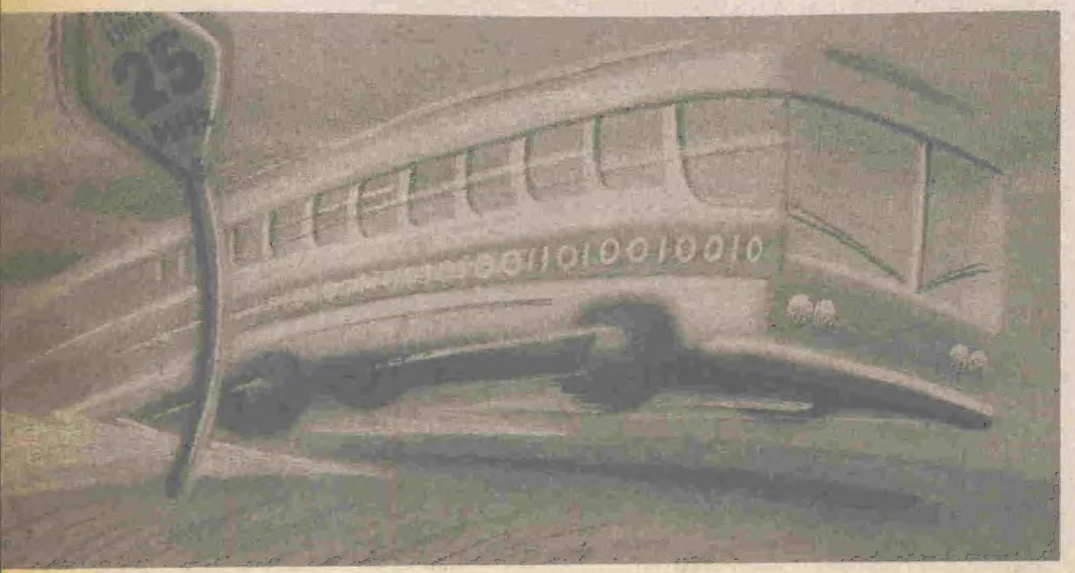
สมชาย นำประเสริฐชัย

เส้นทางของ SBus

ทางเดินของ SBus ยังก้าวต่อไป ทำให้การส่งข้อมูลระดับเวอร์กสเตชันสามารถส่งได้รวดเร็ว



เทคโนโลยีในโลกปัจจุบันก้าวหน้าไปอย่างรวดเร็ว มีการพัฒนากันอยู่ตลอดเวลาเพื่อเพิ่มประสิทธิภาพให้แก่เครื่องจักรหรืออุปกรณ์ต่าง ๆ เครื่องคอมพิวเตอร์ก็จัดว่าเป็นส่วนหนึ่งของเครื่องจักรและอุปกรณ์เหล่านั้น โดยได้พัฒนามาเป็นระยะเวลานานพอสมควร จากระบบที่ใช้ในการคำนวณปัญหาง่าย ๆ จนกระทั่งเป็นระบบที่ใช้วิเคราะห์และตัดสินใจแทนมนุษย์ในปัญหาที่ซับซ้อน สิ่งหนึ่งที่ขาดไม่ได้สำหรับการพัฒนาของคอมพิวเตอร์ก็คือ ประสิทธิภาพของการประมวลผลซึ่งจะต้องถูกต้องแม่นยำและรวดเร็ว จึงพบได้ว่ามีการสร้างชิพที่ใช้ประมวลผล หรือชนิดของซีพียูขึ้นอยู่บ่อย ๆ เพื่อให้มีการทำงานได้ดียิ่งขึ้น แต่อีกสิ่งหนึ่งที่มองข้ามไม่ได้คือ ตัวเชื่อมต่อระหว่างซีพียูกับอุปกรณ์รอบข้างต่าง ๆ เพราะถึงแม้ว่าซีพียูจะประมวลผลเร็วเท่าไรก็ตาม ถ้าไม่สามารถส่งผ่านออกไปได้อย่างรวดเร็วแล้วก็ไม่มีประโยชน์อะไร สิ่งที่เชื่อมหรือเป็นเส้นทางผ่านของข้อมูลก็คือ บัส นั่นเอง ถ้าหากบัสนั้นมีประสิทธิภาพสูงแล้วก็จะเป็นการส่งเสริมการทำงานของคอมพิวเตอร์ด้วย

เอสบัส (SBus) เป็นไอโอบัส ที่ออกแบบและพัฒนาโดยบริษัท ซันไมโครซิสเต็มเพื่อใช้กับเครื่องคอมพิวเตอร์ระดับเวอร์กสเตชันของบริษัทเอง โดย SBus ได้ถูกออกแบบให้มีประสิทธิภาพสูงและใช้โปรโตคอลในการส่งแบบง่าย ๆ และยังได้รักษาคุณสมบัติเดิม ๆ อุปกรณ์

อินพุตเอาต์พุตไว้ อีกทั้งยังเปิดโอกาสให้ผู้อื่นร่วมพัฒนาอีกด้วย ปัจจุบันนี้เครื่องคอมพิวเตอร์ระดับเวอร์กสเตชันส่วนใหญ่ได้ใช้ SBus แล้ว มิใช่มีเพียงแค่เครื่องซันสปาร์ก (Sun Sparestation) เท่านั้น ยังมี Opus, Solbourne และ Mar Microsystems อีกด้วย

จุดมุ่งหมายของ SBus

การส่งข้อมูลในเครื่องคอมพิวเตอร์ระดับเวอร์กสเตชันนั้นต้องการช่วงแถบความถี่สูง บัสที่จะใช้นั้นต้องสามารถส่งผ่านข้อมูลได้อย่างรวดเร็ว เพราะว่าอุปกรณ์หลาย ๆ ประเภทบนเครื่องเวอร์กสเตชันนั้นมีความต้องการ ความสามารถนี้มาก อย่างเช่น โปรเซสเซอร์ประมวลผลสัญญาณดิจิตอล ต้องการอัตราเร็ว 5-20 MBps, FDDI ต้องการในช่วง 12 MBps และยิ่งกว่านั้นยังต้องการซีพียูที่มีประสิทธิภาพสูงอีกด้วย

SBus สามารถส่งข้อมูลได้ในอัตรา 80 MBps ใน burst mode 32 บิต และ 160 MBps ใน burst mode 64 บิต ซึ่งตรงตามความต้องการของเครื่องระดับเวอร์กสเตชัน โดยประสิทธิภาพดังกล่าวเป็นผลสืบเนื่องมาจากสัญญาณนาฬิกาของ SBus ทำงานที่อัตราระหว่าง 16.67 และ 25 เมกะเฮิรตซ์ ประกอบกับการทำงานอย่างซิงโครนัสและการใช้โปรโตคอลที่ไม่ซับซ้อนทำให้การส่งผ่านข้อมูลไม่จำเป็นต้องมีโอเวอร์เฮดมาก

การหน่วงในช่วงรอการส่ง ถ้ากำหนดให้ SBus มีค่าต่ำจะเป็นการเพิ่มประสิทธิภาพให้แก่ระบบ เพราะสามารถทำการเปิดปิดบัสได้อย่างรวดเร็ว ซึ่งหากบัสทำงานช้าหรือมีช่วงเวลาหน่วงมาก บางครั้งการประมวลของซีพียูต้องการข้อมูลบางอย่าง แต่บัสต้องนำมาจากที่ DMA ซึ่งตัวซีพียูจะไม่สามารถทำงานได้จนกระทั่งข้อมูลที่ต้องการนั้นได้รับ ใน SBus การโอนถ่ายข้อมูลจะเสร็จสิ้นภายในช่วง 5 สัญญาณนาฬิกาหรือน้อยกว่านี้ ซึ่งเป็นช่วงเวลาสั้น ๆ และการดำเนินการจะทำขนานกันไป โดยลักษณะนี้เป็นการลดช่วงเวลาหน่วง

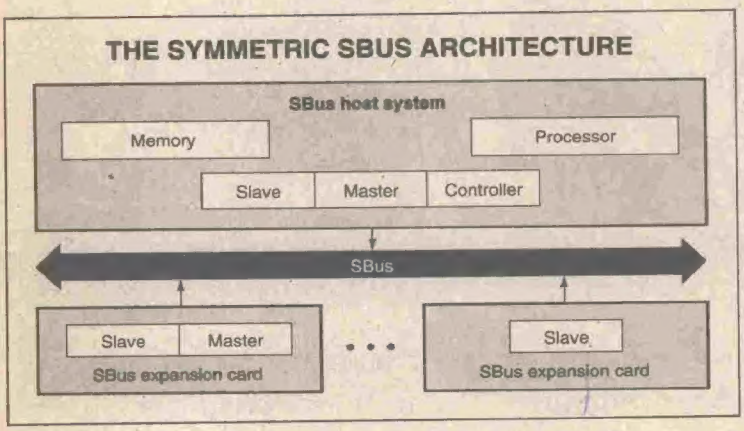
การออกแบบ SBus นั้นมีราคาต่ำ ไม่เหมือนกับโปรเซสเซอร์หรืออุปกรณ์บางอย่างที่ไม่สามารถรอการควบคุมจากบัส เพราะอุปกรณ์เหล่านั้นต้องการหน่วยความจำเพิ่ม เพื่อทำหน้าที่เป็นบัฟเฟอร์ ถ้าหากมีการรอที่นานแล้วขนาดของบัฟเฟอร์ต้องใหญ่ตาม SBus จะลดความต้องการในส่วนที่เป็นบัฟเฟอร์ และจากการที่ใช้หลักการซิงโครนัสแบบง่าย ๆ ทำให้มีความเชื่อถือได้สูง โอกาสเกิดข้อผิดพลาดน้อยมากจึงไม่จำเป็นต้องมีการซิงโครนัสซ้ำ

ผู้พัฒนาและออกแบบ SBus ให้มีการควบคุมเหมือนกับอุปกรณ์ทั่วไป การจัดการ SBus จะขึ้นอยู่กับบัสอื่น เช่น MBus หรือ Future Bus+ สำหรับการเชื่อมโปรเซสเซอร์เข้ากับหน่วยความจำระบบ ซึ่งการใช้บัสร่วมนี้สามารถทำงานขนานกันไปและเป็นการช่วยกันทำงานด้วย

จุดประสงค์หลักอีกประการหนึ่งในการออกแบบ SBus คือ การใช้เทคโนโลยีของ CMOS เพื่อประหยัดพลังงาน โดยขนาดของการ์ด SBus จะมีขนาด 3.3×5.8 นิ้ว และอีกขนาดที่เป็น 2 เท่า จากคุณสมบัติดังกล่าวนี้เหมาะสำหรับเครื่องคอมพิวเตอร์ที่มีขนาดเล็ก เช่น แลปท็อป ซึ่งจะสามารถลดขนาดของพัดลมหรือถอดออกได้เลย เพราะมีการระบายความร้อนต่ำ

SBus มีลักษณะเป็น mezzanine bus คือเป็นการ์ดที่ต่อเชื่อมกับการ์ดหลักหรือการ์ดแม่นั่นเอง อุปกรณ์ของ SBus แบ่งออกเป็น 2 ลักษณะคือ มาสเตอร์ (Master) และสเล็ฟว์ (Slave) โดยส่วนของมาสเตอร์จะมีความซับซ้อนมากกว่า สามารถเริ่มต้นขบวนการส่งได้ การจัดการในบัสทั่วไปส่วนใหญ่จะมีการอ้างถึง DMA (Direct Memory Access) แต่บนตัวของ SBus จะเป็น

การอ้างถึง DVMA (Direct Virtual Memory Access) เพราะส่วนของมาสเตอร์มีการค้างเป็นเวอร์ชวลแอดเดรสเวอร์ชวลเมมโมรีที่เป็นอีกสิ่งหนึ่งที่ส่งเสริมสถาปัตยกรรมของ SBus โอเอสและฟังก์ชันการควบคุมบัสสามารถจัดการกับงานที่ยุ่งยากได้ ลักษณะของ DVMA นั้น ไม่จำเป็นต้องมีฟังก์ชันในการกระจายและรวบรวมข้อมูลเข้าด้วยกัน อีกทั้งลักษณะของเวอร์ชวลแอดเดรสยังช่วยลดปัญหาเกี่ยวกับเรื่องของความยุ่งยากของดิพสวิตช์อีกด้วย และไม่ต้องคำนึงถึงเรื่องของตำแหน่งจริง (physical address)



รูปที่ 1 โครงสร้างสถาปัตยกรรม SBus

Master และ Slave

โครงสร้างสถาปัตยกรรม SBus (แสดงในรูปที่ 1) จะพบว่าประกอบด้วยมาสเตอร์และสเล็วซ์ ตั้งแต่หนึ่งขึ้นไป โดยมาสเตอร์จะเริ่มส่งหลังจากได้ร้องขอและได้รับการควบคุมจากบัส ส่วนสเล็วซ์จะตอบสนอง โดยการเขียนและอ่านข้อมูลเท่าที่ขอจากมาสเตอร์ ส่วนการ์ดขยายอาจมีทั้งคู่หรือมีแค่สเล็วซ์อย่างเดียวก็ได้ ส่วนคอนโทรลเลอร์มีเพียงหนึ่งเดียว ซึ่งจะให้สัญญาณนาฬิกาแก่ SBus และแปลงเวอร์ชวลแอดเดรสอีกทั้งเลือกตัวมาสเตอร์ตัวที่จะดำเนินการส่งต่อไป โดยที่ชื่อและฟังก์ชันต่าง ๆ ของ SBus อยู่ตารางที่ 1

สัญญาณจากบอร์ดแม่จะถูกส่งผ่านไปยังบอร์ดขยายโดยคอนเน็กเตอร์ขนาด 96 pin โดยทุกสัญญาณสามารถใช้การซิงโครนัสแบบง่าย ๆ ยกเว้นสัญญาณ IRQ ทั้งเจ็ดสัญญาณใน SBus จะมีการแยกกันระหว่างข้อมูลกับตำแหน่ง ซึ่งทำให้ง่ายต่อการออกแบบสเล็วซ์ เนื่องจากไม่จำเป็นต้องใช้ demultiplexer และ แลชต์ (latch)

การใช้สัญญาณร่วม

สัญญาณ SBus หลายสัญญาณนั้นใช้ร่วมกัน โดยให้อุปกรณ์หนึ่ง ๆ

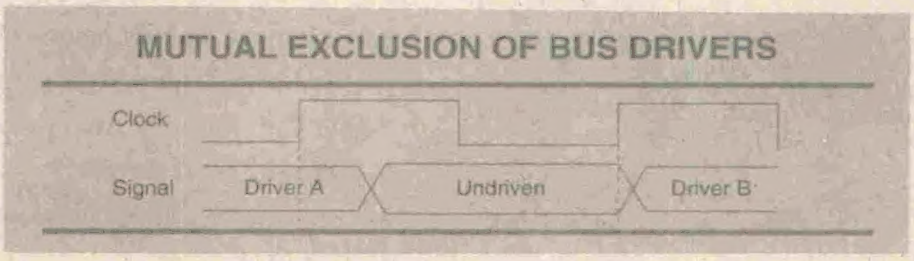
ตารางที่ 1 แสดงชื่อและฟังก์ชันต่าง ๆ ของ SBus

SBUS SIGNALS

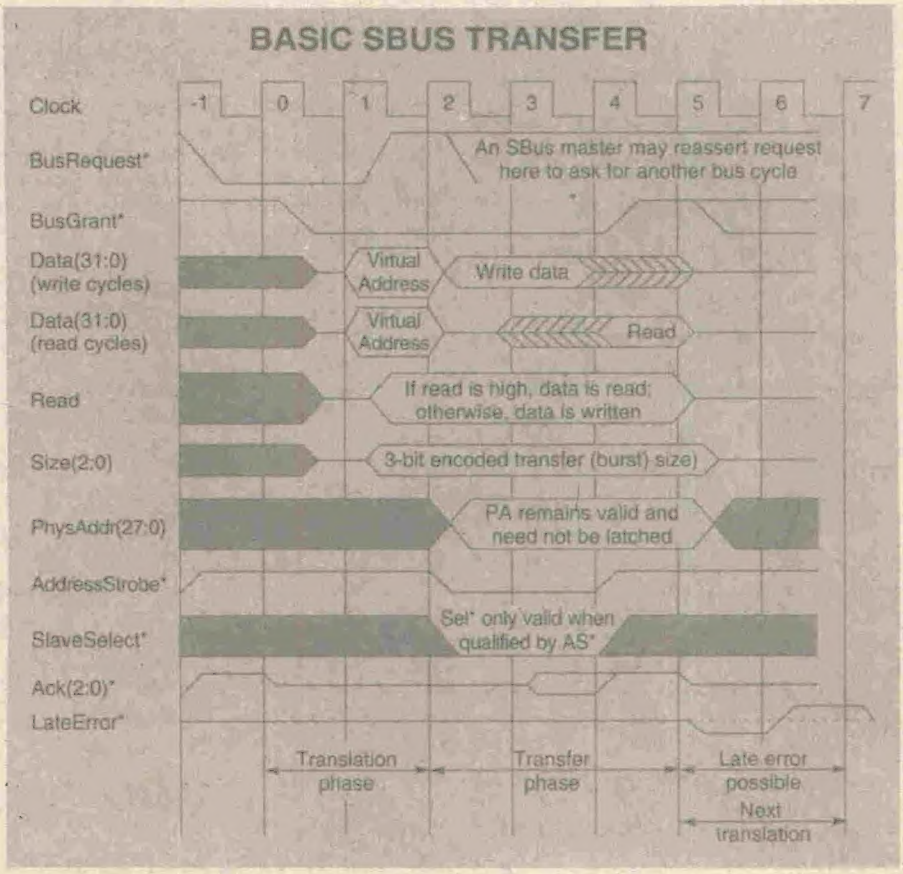
All signals but the seven IRQ signals can be sampled synchronously. The interrupt signals are asynchronous and must be driven with open-collector (or open-drain) outputs. That means that SBus devices can share IRQs.*

Name	Abbreviation	Description	Driven by
PhysAddr(27:0)	PA(27:0)	Physical address	Controller
SlaveSelect*	Sel*	Slave select (one per slave)	Controller
Data(31:0)	D(31:0)	Data	Masters/slaves
Size(2:0)	Siz(2.0)	Transfer size	Masters
Read	Rd	Transfer direction	Masters
Clock	Clk	SBus clock	Controller
AddressStrobe*	AS*	Address strobe	Controller
Ack(2:0)*	Ack(2:0)*	Transfer acknowledgment	Slaves/controller
LateError*	LErr*	Late data error	Slaves
BusRequest*	BR*	Bus request (one per master)	Masters
BusGrant*	BG*	Bus grant (one per master)	Controller
Reset*	Reset*	Reset	Controller
IntReq(7:1)*	IRQ(7:1)*	Interrupt request (open drain)	Slaves
DataParity	DtaPar	Data parity (optional)	Masters/slaves
Ground (7 pins)	Gnd	Ground	Controller
+5 V (5 pins)	+5 V	Power (2 A per slot)	Controller
+12 V	+12 V	Power (30 mA per slot)	Controller
-12 V	-12 V	Power (30 mA per slot)	Controller

นั้นรับสัญญาณในเวลาที่ต่างกัน การครอบครองสามารถเปลี่ยนแปลงได้ในระหว่างการส่งหรือในขณะส่งอ่าน ตัวอย่างเช่น สายของข้อมูลหรือสายดาต้า ในขณะเริ่มต้นของการส่งนั้นตัวมาสเตอร์จะส่งเวอร์ชวลแอดเดรสไปบนสายข้อมูล แต่เมื่อเสร็จสิ้นการส่งแล้วสเล็วซ์จะขับการส่งแทน เมื่อมีการเปลี่ยนแปลงการครอบครองอาจจะทำให้เกิดปัญหาการแย่งใช้บัส หรือที่เรียกว่า บัสไฟท์ (bus fight) ซึ่งเหตุการณ์นี้อาจจะเกิดได้จากระดับหรือการแกว่งของสัญญาณซึ่งเป็นสาเหตุให้เกิดการเหลื่อมล้ำของสัญญาณ ผลที่เกิดคือข้อผิดพลาดหรืออาจจะเกิดความเสียหายให้ไดรเวอร์ก็ได้



รูปที่ 2 แสดงการออกแบบโปรโตคอลของ SBus



รูปที่ 3 แสดงการแลกเปลี่ยนข้อมูลของ SBus

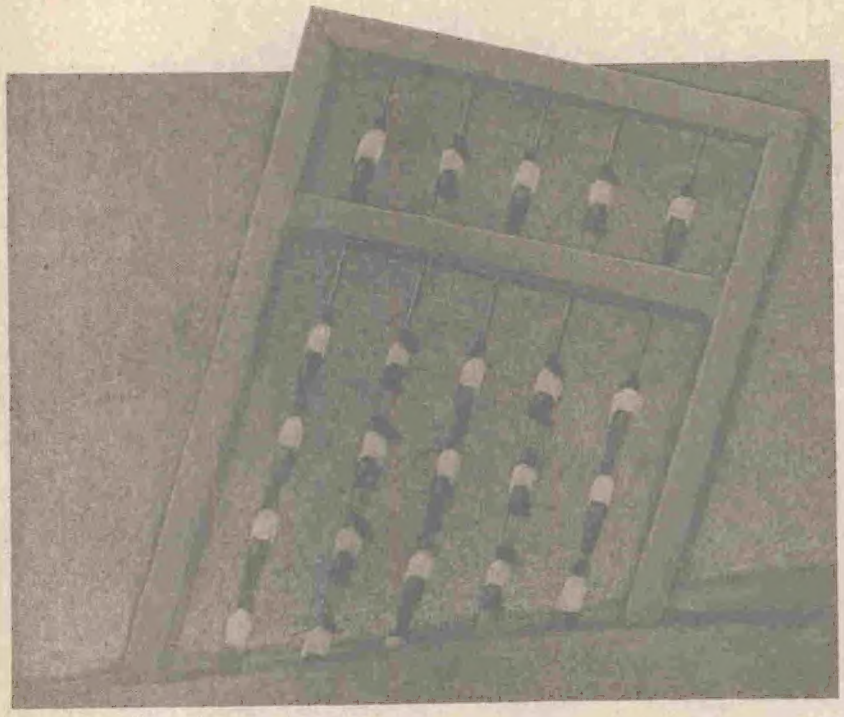
การป้องกันไม่ให้มีเหตุการณ์เช่นนี้เกิดขึ้น คือต้องจัดให้มีเพียงไดรเวอร์เดียวที่ใช้งานเท่านั้น โดยการกำหนดให้ไดรเวอร์จะต้องทำการ disable ก่อนที่ไดรเวอร์ตัวอื่นจะ enable ได้ ดังนั้นโปรโตคอลของ SBus ถูกออกแบบให้มีได้เพียงเอาต์พุตเดียวที่ขับสัญญาณในช่วงหนึ่งคาบของสัญญาณนาฬิกา ตัวอย่างดังรูปที่ 2 ไดรเวอร์ A และ disable ที่ขอบของสัญญาณนาฬิกาที่ 1 และไดรเวอร์ B จะ enable ในสัญญาณนาฬิกาถัดไป

สัญญาณควบคุม Ack และ Late-Error เป็นสัญญาณที่ใช้ร่วมกันโดยจะใช้ขับเมื่อขณะส่งข้อมูลเท่านั้น ส่วนขณะที่ไม่ได้ทำงานจะปล่อยให้ลอยโดยไม่ทราบสภาวะไม่ได้ จะต้องจัดการโดยเก็บสภาวะที่ไม่แน่นอนไว้ จนกระทั่งสัญญาณนั้นถูกขับอีกครั้ง

การแลกเปลี่ยนข้อมูล

รูปแบบการแลกเปลี่ยนข้อมูลของ SBus นั้นสามารถแบ่งออกเป็น 3 ส่วน (ดังแสดงในรูปที่ 3)

1. ส่วนของการจัดการบัส (arbitration phase) ในส่วนนี้บรรดามาสเตอร์ SBus จะต้องแสดงความต้องการในการส่งข้อมูล โดยการส่งสัญญาณ Bus-



Request ซึ่งถ้าหากมีหลายสัญญาณจากหลาย ๆ มาสเตอร์แล้วนั้น ส่วนของ SBus คอนโทรลเลอร์ จะต้องเป็นผู้จัดการในการเลือกให้มาสเตอร์ตัวไหนส่งข้อมูลโดยจะเริ่มส่งข้อมูลได้เมื่อคอนโทรลเลอร์ยืนยันสัญญาณ BusGrant ให้

2. ส่วนของการแปล (Translation Phase) มาสเตอร์จะขับเวอร์ชวลแอดเดรสบนสายข้อมูล และขนาดการลิงค์บนสาย Size และขับทิศทางการส่งบนสาย Read ซึ่งทั้งหมดนี้จะต้องรวดเร็วเสร็จสิ้นภายในช่วงสัญญาณนาฬิกาถัดไป หลังจากนั้น SBus คอนโทรลเลอร์จะทำการแปลงเวอร์ชวลแอดเดรสไปเป็น ฟิสิคัล แอดเดรส ซึ่งจะขับบนสายของ PhysAddr คอนโทรลเลอร์ยังเข้ารหัสตำแหน่งและส่งสัญญาณ SlaveSelect สัญญาณหนึ่งไปยังสเล็วซ์ ซึ่งเป็นเป้าหมาย ทั้งสัญญาณ PhysAddr และ SlaveSelect ที่ถูกขับคอนโทรลเลอร์ส่งสัญญาณ AddressStrobe อีกสัญญาณหนึ่ง เพื่อแสดงความถูกต้องของข้อมูล

3. ส่วนของการแลกเปลี่ยนข้อมูล (Transfer Phase) เป็นส่วนของ SBus ที่มีการแลกเปลี่ยนข้อมูลระหว่างมาสเตอร์กับสเล็วซ์ ในการจัดการการเขียนมาสเตอร์จะมีการเตรียมข้อมูลต่าง ๆ บนสายข้อมูลจนกว่าสเล็วซ์จะพร้อมการรับส่งข้อมูลส่วนของมาสเตอร์ต้องรับประกันว่าข้อมูลนั้นถูกต้อง จนกระทั่งที่ขอบสัญญาณนาฬิกาของสัญญาณ Ack เกิดการเหลื่อมซ้อนกัน แสดงถึงมาสเตอร์สามารถขับข้อมูลสำหรับหนึ่งช่วงสัญญาณนาฬิกาที่ผ่าน AddressStrobe ที่ไม่แน่นอนการจัดการนี้มีประโยชน์เพราะสามารถใช้ AddressStrobe ในการซิงโครนัสในฟังก์ชันการส่งข้อมูล สำหรับการจัดการการอ่าน สเล็วซ์จะแทนข้อมูลที่ถูกต้องลงในสาย Data หลังจากขอบสัญญาณ Ack แล้ว สเล็วซ์จะมีการชี้ข้อผิดพลาดโดยการตอบสนองผิดพลาด หรือขับสัญญาณ Latteerror ซึ่งจะใช้เวลา 2 ช่วงสัญญาณนาฬิกาหลังจากรับสัญญาณ Ack นั้น

สรุป

SBus เป็นบัสที่มีประสิทธิภาพ, ราคาไม่แพง และยังประหยัดพลังงานอีกด้วยเหมาะสำหรับเครื่องคอมพิวเตอร์ที่ต้องการประสิทธิภาพสูง โดยเฉพาะอย่างยิ่งเครื่องระดับเวอร์กสเตชันประเภท RISC ในอนาคตต่อไป SBus คงจะได้รับการพัฒนาและให้การสนับสนุนสิ่งต่าง ๆ มากขึ้น โดยเฉพาะคุณสมบัติอย่างหนึ่งที่ได้น่าสนใจคือ ความสามารถของมาสเตอร์ให้สนับสนุนอุปกรณ์หลายๆ อย่าง อาจจะจัดให้มี DSP หลาย ๆ ตัว ทำงานพร้อม ๆ กัน เพื่อให้การทำมัลติโปรเซสซิงได้ง่าย

เมื่อไม่นานมานี้ได้มีการประกาศชิพที่ใช้กับ SBus คือ Motorola's Goldchip โดยจะทำให้สามารถสร้างการ์ด SBus ได้ง่ายยิ่งขึ้น สถาปัตยกรรมของระบบใหม่ ๆ ได้ออกแบบให้มีหลายสล้อต เพื่อให้ใช้ได้กับหลาย ๆ SBus จากความสามารถนี้ไม่แปลกเลยที่จะพบ SBus บนคอมพิวเตอร์ระบบอื่นที่ไม่ใช้สปาร์ก